Post-Quantum Attribute-Based Encryption: Performance Evaluation and Improvement for Embedded Systems

Pericle Perazzo
Department of Information Engineering
University of Pisa
pericle.perazzo@unipi.it

Michele La Manna
Department of Physics
University of Pisa
michele.lamanna@ing.unipi.it

Francesco Iemma
Department of Information Engineering
University of Pisa
f.iemma1@studenti.unipi.it

Abstract

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) is an encryption paradigm that embeds access control functionalities within ciphertexts. It has shown to be useful for protecting privacy and intellectual property in embedded systems, especially when confidential data is temporarily stored on untrusted cloud or edge servers. However, current CP-ABE ciphers are generally based on pairing mathematics, which is broken if attacked by large-scale quantum computers. Thus, these ciphers will not be secure anymore in the future. In this paper, we focus on a RLWE-based CP-ABE cipher (namely the scheme proposed by Gür et al. in 2019 on IEEE Trans. on Computers), which is believed to be resistant to quantum attacks, so it is a candidate replacement of pairing-based ABE schemes in the future quantum world. Specifically, we measure its performance in terms of processing time and memory with reference to two embedded applications: smart home privacy and automotive FOTA intellectual property protection. We also propose a method to improve the encryption efficiency by dividing it into a slow offline phase and a fast online phase.

1 Introduction

Protection of sensitive data is one of the most critical issues of modern information system. Sensitive data exposure placed third among the most widespread cybersecurity problems according to the latest OWASP Top 10 Project classification [17]. These problems are due to the fact that, while *data in travel* is usually secured by protocols like TLS and DTLS, *data at rest* is typically never encrypted and can be exfiltrated by successful attacks. Data at rest refers here to information stored on any Internet-connected device, for example a cloud server or an edge server. Ideally, access to such data must be possible only by its owner and by whom the owner gave authorization to. Therefore, to address the sensitive data exposure problem we need a way to encrypt data at rest, while possibly making it accessible only to authorized entities.

In 2007, Bethencourt et al. [4] proposed Ciphertext-Policy Attribute-Based Encryption (CP-ABE), which fulfills the aforementioned confidentiality and access control requirements. CP-ABE [19] allows us to encrypt information in such a way that only authorized entities can decrypt it afterwards. To do that, CP-ABE embeds an *access policy* into encrypted data. Decryption keys in turn embed a set of *attributes* that describe the decrypting entity. A given decryption key is capable of decrypting a given ciphertext only if the key's attribute set *satisfies* the ciphertext's access policy. Unfortunately, the majority of CP-ABE schemes existing in the literature do not resist against quantum attacks. This is because they typically employ pairing-based mathematics, which is easily broken by large-scale quantum computers. As a consequence, nowadays embedded security systems based on CP-ABE will become insecure in the future. In the literature there are a few quantum-resistant CP-ABE schemes, which are based on Ring Learning With Errors (RLWE) mathematics, but they consume more resources than quantum-weak CP-ABE schemes.

In this paper, we evaluate the performance of a prominent post-quantum RLWE CP-ABE scheme published in 2019 by Gür et al. [7]. In particular, we measure its performance in terms of processing time and memory with reference to two embedded applications: smart home privacy and automotive FOTA intellectual property protection. We also propose to divide the Gür et al.'s encryption algorithm into an offline and an online phase, which sensibly improves encryption performance.

The rest of the paper is structured as follows. Section 2 presents some related work. Section 3 introduces the necessary preliminary concepts. Section 4 exposes two reference embedded computing applications. Section 5 presents and discusses the experimental results, and it introduces our offline/online encryption technique. Finally, Section 6 concludes the paper.

2 Related Work

Pérez et al. [18] proposed a combined use of symmetric cryptography and CP-ABE to protect privacy sensitive data in smart homes and smart buildings. In the authors' proposal, only authorized services are able to decrypt and access spe-

cific pieces of information in such a way to protect users' privacy, but at the same time the system is sustainable by embedded devices thanks to efficient symmetric cryptography. Authors also demonstrated the system on a real IoT-enabled building. Despite interesting as an application, their proposed system is based on a pairing-based CP-ABE cipher, which does not resist quantum attacks. Therefore, the security it provides is for the present but not for the future.

Baza et al. [3] proposed a peer-to-peer FOTA system for smart cars leveraging blockchain, smart contract, and zk-SNARK technologies to guarantee participation incentives to peers. Their proposed system employs also CP-ABE to protect intellectual property of firmware updates. The general idea is ingenious, but the authors use a quantum-weak CP-ABE cipher, and therefore their security system will not resist future attacks. La Manna et al. [12] evaluated the performance of an automotive FOTA intellectual property protection systems based on CP-ABE on a Xilinx ZCU102 evaluation board, which is representative of the real computing capabilities of modern ECUs. Even here, the employed CP-ABE cipher does not feature post-quantum resistance.

Note that every embedded system described before ([18] for smart home privacy, [3, 12] for automotive FOTA) can be patched to be quantum resistant by using RLWE CP-ABE ciphers in order to remain secure in the future. However, their feasibility evaluation must be redone, since the post-quantum ciphers offer very different performance with respect to their pre-quantum counterparts.

Güneysu et al. [6] and La Manna et al. [13] measure the performance of several post-quantum cryptographic schemes on embedded devices. However, their evaluation is focused exclusively on digital signature schemes, namely GLP, BLISS, and Dilithium in [6] and Dilithium and FALCON in [13]. Digital signatures are of course paramount for authenticating data, but they must be combined with encryption if we want to protect privacy or intellectual property. In this paper we present a performance evaluation of a post-quantum CP-ABE scheme in embedded devices with reference to two example applications: smart home privacy and automotive FOTA intellectual property protection.

Gür et al. [7] proposed a ring-based version of the post-quantum lattice-based CP-ABE scheme by Zhang et al. [22], and also implemented it for the Palisade library and measured its performance. The Gür et al.'s CP-ABE scheme and implementation is the reference post-quantum CP-ABE scheme for the present paper. While the original paper of Gür et al. tested the CP-ABE cipher on a full-resource PC equipped with GPU acceleration, we tested their scheme on a far more constrained device oriented to embedded applications, namely a Raspberry Pi 3 Model B.

3 Preliminaries

3.1 Gür et al.'s CP-ABE Scheme

In this paper we focus on the post-quantum RLWE CP-ABE scheme by Gür et al. [7], which is a ring-based version of a scheme by Zhang et al. [22]. For the sake of brevity, we will not illustrate here the whole scheme but only those aspects that are necessary to explain the experimental results of Section 5. The interested reader can refer to [7] for full details.

Gür et al.'s CP-ABE scheme provides for four algorithms.

1. A Setup algorithm that takes as input a security parameter and the total number of attributes used in the scheme (*attribute universe*) and produces a master key that must be kept secret by the key authority, and some public parameters that must be publicly distributed.
2. A KeyGen algorithm that takes the master key, the public parameters, and a set of attributes (subset of the universe) that describes the decrypting entity and produces a decryption key.
3. An Encrypt algorithm that takes the public parameters, a plaintext to be encrypted, and an access policy and produces a ciphertext.
4. A Decrypt algorithm that takes the public parameters, a ciphertext to be decrypted, and a decryption key and produces a recovered plaintext, or *null* if that decryption key is not authorized to decrypt that ciphertext.

The access policies are composed by a series of AND logical operations between attributes, each of which may be preceded by a NOT operator. The set of the attributes that appear without the NOT inside the policy are said to be *affirmed attributes*. Conversely, those that appear with the NOT are said *negated attributes*. An example of access policy is the following one: "attr1 AND attr2 AND NOT attr3", where "attr1" and "attr2" are affirmed attributes while "attr3" is negated. The attributes of the universe that do not appear in the policy are said *don't-care attributes*, since their logical value does not influence the outcome of the policy.

The master key MSK is composed by a single element:

$$MSK = \{T_A\}. \tag{1}$$

The public parameters MPK are composed by:

$$MPK = \{A, \{B_i^+, B_i^-\}_{\forall i \in \mathcal{X}}, \beta\}, \tag{2}$$

where $\mathcal{X}$ is the attribute universe. Each decryption key SK is composed by:

$$SK = \{\omega_A, \{\omega_i\}_{\forall i \in \mathcal{X}}\}, \tag{3}$$

Finally, each ciphertext CT is composed by:

$$CT = \{C_{0,A}, \{C_{0,i}\}_{\forall i \in \mathcal{W}}, \{C_{0,i}^+, C_{0,i}^-\}_{\forall i \in \mathcal{X} \setminus \mathcal{W}}\}, \tag{4}$$

where $\mathcal{W}$ is the set of attributes that are affirmed or negated in the policy. The set $\mathcal{X} \setminus \mathcal{W}$ represents therefore the don't-care attributes of the policy. The size of all the above cryptographic quantities depends on a *base* parameter (b), which greatly influences the overall performance of the scheme (see Section 5). The Encrypt algorithm is described by Algorithm 1. In the algorithm, μ is the plaintext to be encrypted, $\|\mathcal{X}\|$ is the number of attributes in the universe, $\mathcal{W}^+$ is the set of attributes affirmed in the policy, and $\mathcal{W}^-$ is the set of attributes negated in the policy. The operation $x \leftarrow_U R_q$ means that x is assigned a uniformly random polynomial with Z_q coefficients, whereas the operations $x \leftarrow D_{R,\sigma}$ and $x \leftarrow D_{R^m,\sigma}$ mean that x is assigned respectively a Gaussian distributed random polynomial with Z_q coefficients and σ standard deviation, and an array of m Gaussian distributed random polynomials with Z_q coefficients and σ standard deviation.

Algorithm 1 Encrypt

function ENCRYPT($\mu, MPK, \mathcal{W}$)
  $s \leftarrow_U R_q$
  $e_1 \leftarrow D_{R,\sigma}$
  $c_1 \leftarrow s\beta + e_1 + \mu\lceil \frac{q}{2} \rceil$
  $e_{0,A} \leftarrow D_{R^m,\sigma}$
  $C_{0,A} \leftarrow A^T s + e_{0,A}$
  for $i = 1$ **to** $\|X\|$ **do**
    if $i \in \mathcal{W}^+$ **then**
      $e_{0,i} \leftarrow D_{R^m,\sigma}$
      $C_{0,i} \leftarrow (B^+)^T s + e_{0,i}$
    else if $i \in \mathcal{W}^-$ **then**
      $e_{0,i} \leftarrow D_{R^m,\sigma}$
      $C_{0,i} \leftarrow (B^-)^T s + e_{0,i}$
    else
      $e^+_{0,i}, e^-_{0,i} \leftarrow D_{R^m,\sigma}$
      $C^+_{0,i} \leftarrow (B^+)^T s + e^+_{0,i}$
      $C^-_{0,i} \leftarrow (B^-)^T s + e^-_{0,i}$
    end if
  end for
  $CT \leftarrow \{\mathcal{W}, C_{0,A}, \{C_{0,i}\}_{i\in\mathcal{W}}, \{C^+_{0,i}, C^-_{0,i}\}_{i\in X\setminus\mathcal{W}}, c_1\}$
  return CT
end function

3.2 Palisade Library

Palisade[1] is an open-source C++ library that implements several lattice-based cryptography building blocks and schemes. Though it is mainly oriented to homomorphic encryption, it also provides some support for post-quantum public-key encryption and signature, proxy re-encryption, multiparty computation, identity-based encryption and attribute-based encryption. To the best of our knowledge and at the time of writing, Palisade is the only library supporting CP-ABE enjoying post-quantum security. The authors of the CP-ABE cipher that we employ in this paper [7] implemented their scheme by means of the Palisade building blocks. Palisade also offers some predefined parameters, which comply to the HomomorphicEncryption.org standards [1].

4 Reference Applications

The application of CP-ABE ciphers to protect privacy and intellectual property in embedded systems has been already considered in the literature in different scenarios, for example industrial automation [11], smart grid [16], smart city [20], e-health [5, 23], smart vehicles [3] etc. In this paper, we focus on two particular applications of CP-ABE in embedded systems: privacy protection in smart homes [21], and intellectual property protection in automotive FOTA [12].

4.1 Privacy Protection in Smart Homes

Home automation [10] refers to the capability of controlling and monitor home devices, such as lighting, climate and HVACs, infotainment systems, cameras and alarm systems, various appliances, and so on. In general, such sensors and actuators are connected to the Internet through some smart home hub called *gateway*, and they can interact remotely with smartphones or other mobile devices. The amount and the nature of data produced by such smart home devices rise high privacy concerns. Think for example of IP cameras that sends recorded videos to the gateway and from there to some cloud/edge server that afterwards serves the videos to authorized people. Different people could have different authorizations, for example the house tenant could be able to see all the videos, while the landlord or a guest could be able to see only some of them. Securing the Internet connection channel (for example with TLS connection) is necessary but may be insufficient to provide complete protection. Indeed, video material is unprotected when at rest on intermediate cloud/edge servers, which may be untrusted. The best situation for privacy is when videos are protected end-to-end from the IP camera to the authorized devices that downloads them from the cloud/edge and reproduce them. This can be done with CP-ABE, while at the same time guaranteeing access control policies to authorize different people to decrypt different videos. Figure 1 shows an example of such an application.

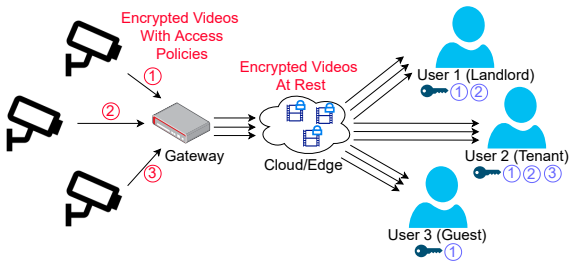

Figure 1. Smart home application example.

In the figure, the red circled numbers identify the video sources, while the blue circled numbers express the authorization of the different users. The landlord (User 1) can decrypt all videos except those recorded by Camera 3. The tenant (User 2) can decrypt every video, while the guest (User 3) can decrypt only videos recorded by Camera 1.

4.2 Intellectual Property Protection in Automotive FOTA

Nowadays, vehicles are increasingly becoming "smart", in the sense that they rely on many ECUs and electronic components to improve the driver's experience with new features. However, as the Hyppönen's law predicts, "smart means exploitable" [14]. It is likely that the increased complexity of the software installed on modern vehicles will lead to many bugs and security vulnerabilities, which in turn will require complex software update management systems, like those we already see on PCs and mobile devices. A promising solution for the aforementioned problems is adopting the Firmware-Over-The-Air (FOTA) paradigm [2, 8]. The basic idea is to leverage the Internet connection available in the modern infotainment ECUs in order to receive software updates for the other ECUs of the vehicle. During this process, protecting the privacy of the software vendor in terms of Intellectual Property (IP) is paramount. Without any protection, competitors can easily capture and reverse engineer the

[1] https://palisade-crypto.org/

software in order to learn industrial secrets. Even in this application, securing the infotainment communication channel is necessary but insufficient to provide complete protection. Indeed, confidentiality is not guaranteed when the software update is at rest on intermediate cloud/edge servers, which may be untrusted. Moreover, confidentiality is not guaranteed even when the software update is at rest in the infotainment ECU itself, which is the most exposed ECU to cyberattacks [9]. Ideally, software updates should be protected end-to-end from the software producer to the final target ECU. Figure 2 shows an example of such an application.

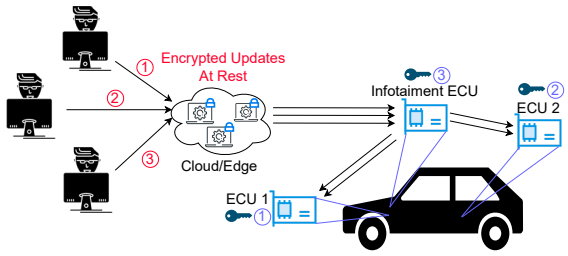

Figure 2. Automotive FOTA application example.

Similarly to the previous section, the red circled numbers identify the software sources, while the blue circled numbers express the authorization of the different ECUs inside the cars.

5 Experimental Results

In order to test the Gür et al.'s CP-ABE scheme [7] we have developed a benchmark program that measures the processing time of each algorithm of the cryptographic scheme. The benchmark program also measures the sizes of each cryptographic quantity involved in the operations, namely the master key, the public parameters, the decryption keys and the ciphertexts. We set a universe of 32 attributes, which should be fine for the majority of embedded applications. We assume to employ a digital envelope technique, that is we encrypt a random symmetric key with CP-ABE, and then we encrypt the actual plaintext with such a symmetric key. To resist to Grover quantum attack against the symmetric key, we used keys with double size with respect to the target security level, i.e. 256 bits for 128-bit security level, 384 bits for 192-bit security level, and 512 bits for 256-bit security level. Note that at the time of writing there are no standardized symmetric-key ciphers employing keys bigger than 256 bits, even if some proposals are present in the literature [15]. We tested three different security levels (128 bits, 192 bits, 256 bits), adopting the three sets of default parameters provided by the Palisade library.

Since the ciphertext contains at least an element for each attribute in the universe (either affirmed, negated, or don't-care), its size should grow linearly with the universe size. However, using different access policies leads to different performance because, as we can see from Algorithm 1, when the attribute is a don't-care either $C^+_{0,i}$ and $C^-_{0,i}$ are computed and put inside the ciphertext. On the other hand, when we have an affirmed or negated attribute, $C^+_{0,i}$ and $C^-_{0,i}$ are replaced by a single $C_{0,i}$ element. Since $\{C^+_{0,i}, C^-_{0,i}\}$ is bigger than $C_{0,i}$, a ciphertext with many don't-care attributes in the policy should be bigger than a ciphertext with few of them. To capture the effect of the access policy on performance we tested three types of access policy, namely:

- Best-case access policy, in which there are no don't-care attributes;
- Average-case access policy, in which 16 attributes are don't-care and the other 16 are affirmed or negated;
- Worst-case access policy, in which all the attributes but one are don't-care.

We set the worst-case access policy to contain at least one affirmed or negated attribute because otherwise such a policy could meaninglessly authorize everyone to decrypt the ciphertext.

All the experiments are performed on a Raspberry Pi 3 Model B (Quad Core 1.2GHz 64bit CPU, 1GB RAM). Finally, we performed 30 repetitions for each experiment in order to get more statistically meaningful results.

Figure 3 shows the processing time of the various CP-ABE algorithms with different security levels.

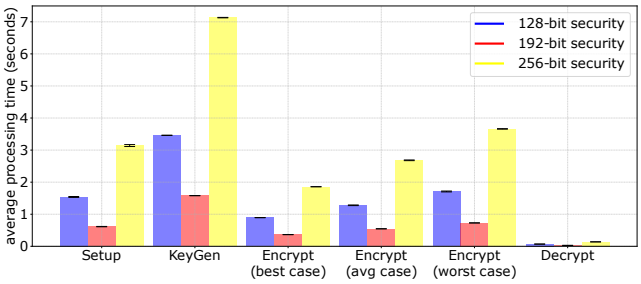

Figure 3. Processing time of CP-ABE algorithms. 99%-**confidence intervals are displayed in error bars.**

From our experiments we noticed that the access policy type impacts on the encryption efficiency but it does not influence the other algorithms. Therefore, for all the other algorithms we report the results of the average-case policies only.

The results show that the 192-bit security parameters offer unexpectedly better performance than the 128-bit ones. This is due to the base used in this parameters. Indeed performance heavily depends on the base: the bigger it is, the better is the performance from the viewpoint of the processing time. The base of the 192-bit parameters is $b_{192} = 128$ whereas that of the 128-bit parameters is $b_{128} = 2$. This suggests us that the 128-bit Palisade parameters are sub-optimal and could be improved by successive versions of the library. Another observation that we can make from Figure 3 is that the encryption time is much longer than the decryption time. Decryption is already much efficient (69 milliseconds for 128-bit security), and this makes the scheme ready to be applied in the automotive FOTA application scenario described in Section 4.2. Indeed, in that scenario the device that encrypts (i.e., the software producer) has plenty of resources, while the devices that decrypt (i.e., the vehicle's ECUs) have scarce resources. However, the inefficient encryption algorithm could be a problem when we apply the scheme in a scenario where devices that produce data have scarce resources,

for example the smart home scenario described in Section 4.1. In the following we propose a way to overcome this limitation.

The encryption algorithm proposed by Gür et al. (Algorithm 1) generates a series of errors. In particular, it generates an error and an error vector (e_1 and $e_{0,A}$, Lines 3 and 5), one error vector for each affirmed or negated attribute ($e_{0,i}$, Lines 9 and 12), and two error vectors for each don't-care attribute ($e^+_{0,i}$ and $e^-_{0,i}$, Line 15).

Since generating such error vectors is the bottleneck of the encryption algorithm, our idea is to precompute them all during an *offline encryption phase*. Such a phase can be done in advance by a trusted resourceful device, or possibly by the embedded device itself during charging time. We call *precomputed encryption material* (E) the set of error vectors produced by the offline encryption phase. Afterwards, all the remaining computations needed to encrypt are done in an *online encrypt phase*, which is enough lightweight to be performed by the embedded device itself. Also, we want the offline encrypt phase to be independent of the access policy, therefore we need to generate a set of error vectors that fits all the possible access policies. We obtain this by generating $e^+_{0,i}$ and $e^-_{0,i}$ for each attribute of the universe except one, for which we generate only one error vector $e_{0,i}$. Again, we assume here that the worst-case access policy contains at least one affirmed or negated attribute because otherwise such a policy could meaninglessly authorize everyone to decrypt the ciphertext. During the online phase, for each affirmed (negated) attribute, we will discard the corresponding $e^-_{0,i}$ ($e^+_{0,i}$) element. The offline encrypt phase is described in Algorithm 2.

Algorithm 2 Offline Encrypt

1: **function** OFFLINEENCRYPT
2: $e_1 \leftarrow D_{R,\sigma}$
3: $e_{0,A} \leftarrow D_{R^m,\sigma}$
4: **for** $i = 1$ **to** $\|\mathcal{X}\| - 1$ **do**
5: $e^+_{0,i}, e^-_{0,i} \leftarrow D_{R^m,\sigma}$
6: **end for**
7: $e_{0,\|\mathcal{X}\|} \leftarrow D_{R^m,\sigma}$
8: $E \leftarrow \{e_1, e_{0,A}, \{e^+_{0,i}, e^-_{0,i}\}_{i=1,..,\|\mathcal{X}\|-1}, e_{0,\|\mathcal{X}\|}\}$
9: **return** E
10: **end function**

In order to prove the effectiveness of this solution we performed another series of experiments with the same settings of before but dividing the encryption in the offline and online phases. Figure 4 shows the sizes of the master secret, the public parameters, the decryption keys, the precomputed encryption material, and the ciphertexts.
From the experimental data we observed that the sizes do not change from one repetition to another. For this reason we omit the confidence intervals in the figure.

First of all it should be noted that all the cryptographic quantities with 192-bit security parameters are smaller in size than their counterparts with 128-bit security parameters. This is again due to the used base, and it further proves the sub-optimality of the Palisade standard 128-bit parameters. It should also be noted that the master key has a negligible size, since it does not depend on the size of the attribute universe. On the contrary, the sizes of the public parameters and of the decryption keys are quite big, because they linearly grows with the size of the universe. Note also that the precomputed encryption material is ~ 40 MB for the 256-bit security level. Considering that we need a piece of precomputed encryption material for each ciphertext, this seems beyond the possibilities of nowadays embedded devices and protocols. However, since quantum attacks are not an immediate threat, we foresee that future embedded devices and protocols will manage such quantities. Moreover, we can employ symmetric-key cryptography to sensibly decrease the impact on memory and bandwidth. For example an embedded device can encrypt all the data having the same access policy with a single symmetric key and then encrypt such a key with CP-ABE, like proposed in [11].

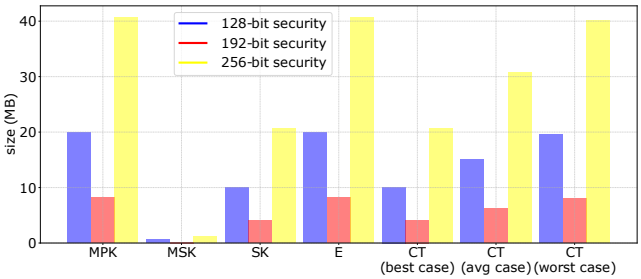

Figure 4. Size of cryptographic quantities produced by the CP-ABE scheme.

Figure 5 shows the processing time of the offline and online encryption phases with different security levels.

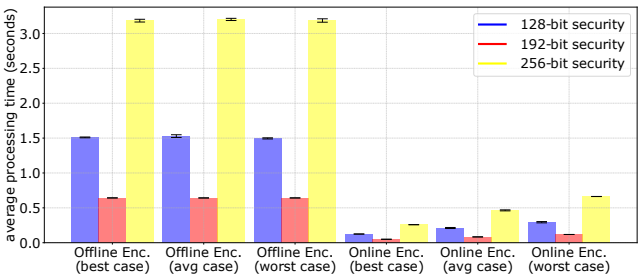

Figure 5. Processing time of offline/online encrypt algorithm with different types of policies

It can be noted that the online encryption phase is sensibly quicker than the complete encryption algorithm. For example, with 128-bit security and average-case access policies, a complete encryption is 1.27 seconds while an online encryption phase is only 211 milliseconds. This proves that our technique is effective for embedded devices, providing that they have enough space to store the precomputed encryption material.

6 Conclusions

In this paper, we evaluated the performance of a state-of-the-art post-quantum RLWE CP-ABE scheme published in 2019 by Gür et al. [7]. In particular, we measured its encryption and decryption performance in terms of processing

time and memory with reference to two embedded applications: smart home privacy and automotive FOTA intellectual property protection. Our experiments suggest that current CP-ABE implementations are ready to be applied in the automotive FOTA application scenario, but they are more problematic for the smart home scenario due to the encryption inefficiency. We thus proposed to divide the Gür et al.'s encryption algorithm into an offline and an online phase, which sensibly improves encryption performance in the smart home scenario.

7 References

[1] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, S. Lokam, D. Micciancio, D. Moody, T. Morrison, A. Sahai, and V. Vaikuntanathan. Homomorphic encryption security standard. Technical report, HomomorphicEncryption.org, Toronto, Canada, November 2018.

[2] N. Asokan, T. Nyman, N. Rattanavipanon, A.-R. Sadeghi, and G. Tsudik. ASSURED: Architecture for secure software update of realistic embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2290–2300, 2018.

[3] M. Baza, M. Nabil, N. Lasla, K. Fidan, M. Mahmoud, and M. Abdallah. Blockchain-based firmware update scheme tailored for autonomous vehicles. In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–7, 2019.

[4] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy (SP '07)*, pages 321–334, 2007.

[5] K. Edemacu, H. K. Park, B. Jang, and J. W. Kim. Privacy provision in collaborative ehealth with attribute-based encryption: Survey, challenges and future directions. *IEEE Access*, 7:89614–89636, 2019.

[6] T. Güneysu, M. Krausz, T. Oder, and J. Speith. Evaluation of lattice-based signature schemes in embedded systems. In *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 385–388, 2018.

[7] K. D. Gür, Y. Polyakov, K. Rohloff, G. W. Ryan, H. Sajjadpour, and E. Savaş. Practical applications of improved gaussian sampling for trapdoor lattices. *IEEE Transactions on Computers*, 68(4):570–584, 2019.

[8] T. Karthik, A. Brown, S. Awwad, D. McCoy, R. Bielawski, C. Mott, S. Lauzon, A. Weimerskirch, and J. Cappos. Uptane: Securing software updates for automobiles. In *International Conference on Embedded Security in Car*, pages 1–11, 2016.

[9] S. K. Khan, N. Shiwakoti, P. Stasinopoulos, and Y. Chen. Cyber-attacks in the next-generation cars, mitigation techniques, anticipated readiness and future directions. *Accident Analysis Prevention*, 148:105837, 2020.

[10] M. K. Kuyucu, Bahtiyar, and G. İnce. Security and privacy in the smart home: A survey of issues and mitigation strategies. In *2019 4th International Conference on Computer Science and Engineering (UBMK)*, pages 113–118, 2019.

[11] M. La Manna, P. Perazzo, M. Rasori, and G. Dini. fABElous: An attribute-based scheme for industrial internet of things. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 33–38, 2019.

[12] M. La Manna, L. Treccozzi, P. Perazzo, S. Saponara, and G. Dini. Performance evaluation of attribute-based encryption in automotive embedded platform for secure software over-the-air update. *Sensors*, 21(2), 2021.

[13] M. L. Manna, P. Perazzo, L. Treccozzi, and G. Dini. Assessing the cost of quantum security for automotive over -the-air updates. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6, 2021.

[14] H. Mikko, L. Nyman, et al. The Internet of (vulnerable) things: On Hyppönen's law, security engineering, and IoT legislation. *Technology Innovation Management Review*, 2017.

[15] A. Moh'd, Y. Jararweh, and L. Tawalbeh. AES-512: 512-bit advanced encryption standard algorithm design and evaluation. In *2011 7th International Conference on Information Assurance and Security (IAS)*, pages 292–297, 2011.

[16] J. M. Navya, H. A. Sanjay, and K. Deepika. Securing smart grid data under key exposure and revocation in cloud computing. In *2018 3rd International Conference on Circuits, Control, Communication and Computing (I4C)*, pages 1–4, 2018.

[17] OWASP. *OWASP Project Top 10*, 2021. https://owasp.org/www-project-top-ten/.

[18] S. Pérez, J. L. Hernández-Ramos, S. N. Matheu-García, D. Rotondi, A. F. Skarmeta, L. Straniero, and D. Pedone. A lightweight and flexible encryption scheme to protect sensitive data in smart building scenarios. *IEEE Access*, 6:11738–11750, 2018.

[19] M. Rasori, M. L. Manna, P. Perazzo, and G. Dini. A survey on attribute-based encryption schemes suitable for the internet of things. *IEEE Internet of Things Journal*, 9(11):8269–8290, 2022.

[20] M. Rasori, P. Perazzo, and G. Dini. ABE-Cities: An attribute-based encryption system for smart cities. In *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 65–72, 2018.

[21] S. Sicari, A. Rizzardi, G. Dini, P. Perazzo, M. La Manna, and A. Coen-Porisini. Attribute-based encryption and sticky policies for data access control in a smart home scenario: a comparison on networked smart object middleware. *International Journal of Information Security*, 20(5):695–713, 2021.

[22] J. Zhang, Z. Zhang, and A. Ge. Ciphertext policy attribute-based encryption from lattices. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, page 16–17, New York, NY, USA, 2012. Association for Computing Machinery.

[23] Y. Zhang, D. Zheng, and R. H. Deng. Security and privacy in smart health: Efficient policy-hiding attribute-based access control. *IEEE Internet of Things Journal*, 5(3):2130–2145, 2018.